

## REMARKS

Claims 1-33 remain in the application for consideration. In view of the following remarks, Applicant respectfully requests withdrawal of the rejections and forwarding of the application onto issuance.

### Drawing Objections

The drawings are objected to as having unacceptable margins. Particularly, Figs. 1, 3, and 6 are indicated to have unacceptable bottom margins. Applicant submits herewith corrected drawings thereby overcoming the objections.

### Specification

On page 8 under the "Exemplary Architecture" section, the Office indicates that "parser 18" does not match up with a numerical designator in Fig. 2. Submitted herewith is a revised Fig. 2 which contains the numerical designator for the parser element. Applicant thanks the Examiner for the Examiner's attention to detail.

On page 15, an inconsistency in the numerical designation associated with the node factory was observed by the Office. Applicant has amended the specification to correct this oversight. Again, Applicant thanks the Examiner.

### § 112 Rejections

Claims 13 and 23 are rejected under 35 U.S.C. §112, second paragraph as being vague and indefinite as to what steps would be included in the computer program. Applicant disagrees. Claim 13 depends from claim 1 which recites:

- defining a plurality of states, individual states being associated with individual elements of an XML data stream;

- associating one or more rules with each state;
- receiving an XML data stream;
- evaluating the XML data stream against one or more of the rules for individual elements contained in the XML data stream; and
- disregarding associated portions of the XML data stream if any of the rules that are associated with those portions are violated.

Applicant submits that there is nothing unclear with respect to which steps are performed by the recited computer program. In this claim, all of the steps are to be performed by the computer program. This does not mean or imply, however, that the computer program does not receive input from a source such as a human operator. This claim merely captures the functionality that is performed by the computer program. The same can be said of claim 23. Given this, Applicant submits that there is nothing unclear about either of these claims and thus traverses the Office's rejections.

### **The §103 Rejections**

Claims 1-4, 10-13, 30 and 31 stand rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 6,446,256 to Hyman in view of a document to Lyubashevskiy entitled "VT420 Parser".

Claims 5-9, 32 and 33 stand rejected under §103 as being unpatentable over Hyman and Lyubashevskiy, in further view of a document entitled "How to implement Web-based Groupware Systems based on WebDAV" to Dridi and Neumann (hereinafter referred to as "Dridi").

Claims 14, 17 and 23 stand rejected under §103(a) as being unpatentable over Hyman.

Claims 15 and 16 stand rejected under §103(a) as being unpatentable over Hyman in view of U.S. Patent No. 6,411,974 to Graham.

1 Claims 18-22 stand rejected under §103(a) as being unpatentable over  
2 Hyman in view of Dridi.

3 Claims 24-27 and 29 stand rejected under §103(a) as being unpatentable  
4 over Hyman in view of a document entitled "Document Object Model (DOM)  
5 Level 2 Specification, W3C Working Draft (hereinafter referred to as "DOM").

6 Claim 28 stands rejected under §103(a) as being unpatentable over Hyman  
7 in view of DOM and in further view of Dridi.

### 8 9 **The Office's Rejections and The References**

10 The Office has failed to make out a prima facie case of obviousness and, for  
11 the reasons set forth below, Applicant traverses the Office's rejections. Before  
12 discussing, in detail, how the Office has failed to make out a prima facie case of  
13 obviousness, the following discussion of the Hyman and Lyubashevskiy  
14 references is provided.

### 15 16 **The Hyman Reference**

17 Hyman's disclosure pertains to extensions of parsable structures. In  
18 accordance with Hyman's disclosure, a bilateral interface may comprise an object  
19 interface and, optionally, a language interface. The object interface comprises at  
20 least an object-side object interface to be included in an external object, and a  
21 structure-side object interface to be included in a parsable structure. When  
22 executed, an object initialization routine included in the object-side object  
23 interface provides an identification of the structure-side object interface to the  
24 external object. Conversely, a structure initialization routine included in the  
25 structure-side object interface provides an identification of at least one object  
interaction routine to the external object. The external object can then invoke the

1 object interaction routine to interact with the parsable structure. The language  
2 interface is similarly constructed.

3 In connection with its disclosure, Hyman discusses general notions  
4 associated with Extensible Markup Language (XML). For example, in the  
5 "Background" section, Hyman describes that an XML parser can be used to break  
6 down an XML document into a representation that a computer can understand.  
7 Further, Hyman describes generally the operation of an XML parser which, as  
8 noted in column 7, lines 12-14, "reads through the XML document and from that  
9 creates an internal data structure called the XML DOM tree, like that shown in  
10 Fig. 4."

### 11 12 **The Lyubashevskiy Reference**

13 Lyubashevskiy appears to be an excerpt from a parser manual that states,  
14 generally, the function of a parser which is to "parse the data stream by identifying  
15 the ASCII characters and terminal commands and reporting them to the higher-  
16 level code." (See, Lyubashevskiy, paragraph 2). Further on in this reference,  
17 Lyubashevskiy simply states that "[a]n important issue is the integration of the  
18 parser in an application the way that allows parsing to occur while the data is  
19 being received. To satisfy this scheme, the parser has been implemented as a  
20 finite state machine that preserves its state between the calls."

### 21 22 **The Claims**

23 **Claim 1** recites a method of parsing an Extensible Markup Language  
24 (XML) data stream. The recited method comprises:  
25

- defining a plurality of states, *individual states being associated with individual elements of an XML data stream*;
- associating one or more rules with each state;
- receiving an XML data stream;
- evaluating the XML data stream against one or more of the rules for individual elements contained in the XML data stream; and
- disregarding associated portions of the XML data stream if any of the rules that are associated with those portions are violated.

In making out the rejection of this claim, the Office argues that Hyman discloses that when an XML document has an associated schema the parser will make sure that the document follows the rules of the schema. The Office apparently equates this disclosure to the “receiving” and “evaluating” acts recited above. The Office admits that Hyman does not disclose the act of “disregarding associated portions of the XML data stream if any of the rules that are associated with those portions are violated”, but apparently equates this to the notion disclosed in Hyman that if a document does not follow the rules of the schema an error will occur. (See, Office Action page 3).

Based on this disclosure in Hyman, the Office concludes that it would be obvious to have “combined the receiving, associating, and evaluating a data stream of Hyman and disregard associated portions of the XML data stream, which in turn would have streamlined the parsing activities.”

The Office’s rejection and rationale is unclear to Applicant.

The Office then states that Hyman does not disclose “defining a plurality of states, which are associated with individual elements of an XML data stream.” Applicant agrees. The Office then relies on Lyubashevskiy which discloses simply implementing a parser as a finite state machine that preserves its state between calls, and based on this, the Office argues that it would be obvious to implement the finite state machine of Lyubashevskiy to define a plurality of states

1 and associate these states with individual elements of an XML data stream into the  
2 above mentioned parsing method of Hyman to provide a more efficient schema.

3 This rejection and rationale is, at best, unclear to Applicant. Neither  
4 Hyman nor Lyubashevskiy taken singly or in combination with one another  
5 disclose or suggest the subject matter of this claim. The Office has failed to make  
6 out a case of prima facie obviousness for a number of different reasons not least of  
7 which include: (1) that the Office has misinterpreted the cited references; (2) that  
8 the Office has ascribed properties to the references which they simply do not have;  
9 and (3) that the Office has failed to provide a motivation to combine these  
10 misinterpreted references that is *stated with particularity*.

11 Specifically, consider the first recited element of claim 1:

12 *defining a plurality of states, individual states being associated with*  
13 *individual elements of an XML data stream.*  
14

15 Lyubashevskiy simply discloses implementing a parser as a finite state  
16 machine that preserves its state between calls. This disclosure falls far short of  
17 meeting this claim element, for there is no mention whatsoever of “defining a  
18 plurality of states” where individual states are “associated with individual  
19 elements of an XML data stream.” Thus, Lyubashevskiy is completely lacking in  
20 this regard, and such shortfall cannot be made up by ascribing properties to this  
21 reference which it simply does not have. Accordingly, for at least this reason,  
22 claim 1 is allowable.

23 Claim 1 further recites “associating one or more rules with each state”. As  
24 Lyubashevskiy does not disclose or suggest defining a plurality of states that are  
25 associated with individual elements of an XML data stream, it is virtually  
impossible for Lyubashevskiy or Hyman to disclose or suggest associating any

1 rules with any such states as contemplated in claim 1. Accordingly, for this  
2 additional reason, claim 1 is allowable.

3 Claim 1 further recites “evaluating the XML data stream against one or  
4 more of the rules for individual elements contained in the XML data stream.” As  
5 Lyubashevskiy does not disclose or suggest associating one or more rules with  
6 each “state” as that term is used in claim 1, it is virtually impossible for it or  
7 Hyman to disclose or suggest evaluating the XML data stream against one or more  
8 of the rules for individual elements contained in the XML data stream.  
9 Accordingly, for this additional reason, claim 1 is allowable

10 As neither reference teaches nor suggests the subject matter discussed  
11 above, it is virtually impossible for the references to disclose or suggest the last  
12 recited act of “disregarding associated portions of the XML data stream if any of  
13 the rules that are associated with those portions are violated.”

14 Accordingly, for any and/or all of the individual reasons mentioned above,  
15 claim 1 is allowable.

16 **Claims 2-13** depend either directly or indirectly from claim 1 and are  
17 allowable as depending from an allowable base claim. These claims are also  
18 allowable for their own recited features which, in combination with those recited  
19 in claim 1, are neither disclosed nor suggested by the references of record either  
20 singly or in combination with one another. In addition, as these claims are  
21 allowable, Neumann is not seen to add anything of significance to the rejection of  
22 claims 5-9.

23 **Claim 14** recites a method of parsing an Extensible Markup Language  
24 (XML) data stream. The recited method comprises:

- 25
- defining a schema module that is associated with an HTTP request type that is received from a client, the schema module having a

function that determines whether an XML data stream conforms to a given schema that is associated with the HTTP request type;

- evaluating an XML data stream with the schema module; and
- disregarding a portion of the XML data stream if it does not conform to the given schema.

In making out this rejection, the Office argues that Hyman discloses employing XML schemas and that a parser will make sure that the document follows the rules of the schema. The Office then argues that Hyman discloses well known protocols and states that its invention can be employed in a client-server configuration. The Office then apparently equates this to the recited acts of “defining a schema module...” and “evaluating an XML data stream....”

The Office’s rejection is unclear and Applicant submits that the Office has completely disregarded features that are recited in this claim. This claim recites that a schema module is defined and *associated* with an HTTP request type. Nowhere in any of the cited references is there any disclosure or suggestion of a schema module that is associated with an HTTP request type. This feature or anything remotely resembling it is completely missing from any of the references. Accordingly, for at least this reason, this claim is allowable.

As this recited feature is entirely missing from the cited references, it is virtually impossible for the references to disclose or suggest the remaining elements of this claim. The Office has failed to establish a prima facie case of obviousness and, accordingly, this claim is allowable.

**Claims 15-23** depend either directly or indirectly from claim 14 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 14, are neither disclosed nor suggested by the references of record either singly or in combination with one another. In addition, as these claims are



allowable, neither Graham nor Neumann are seen to add anything of significance to the rejection of claims 15-16, and 18-22 respectively.

**Claim 24** recites an Extensible Markup Language (XML) parsing system comprising:

- a parser configured to receive an XML data stream and generate a series of calls as it parses the XML data stream;
- a node factory communicatively associated with the parser and configured to receive the parser's calls and responsive thereto construct a representation of the XML data stream that the parser is parsing; and
- a schema module communicatively associated with the node factory and configured to evaluate the node factory's representation of the XML data stream and determine whether it conforms to a known schema.

In making out this rejection of this claim, the Office essentially argues that Hyman meets all of the elements of this claim except for the recited node factory. The Office then relies on DOM and argues that the disclosed node iterators are equivalent to the node factory. Applicant disagrees. The recited node factory receives the parser's calls and responsive thereto constructs a representation of the XML data stream. DOM's disclosed node iterators do not appear to build anything responsive to receiving a call from a parser. Rather, the node iterators are used to "step through a set of nodes". (See DOM, page 120, section 6.2, first paragraph). The nodes that are stepped through by DOM's node iterators do not appear to be constructed responsive to receiving a parser's calls. Accordingly, the Office has mischaracterized this reference. As such, this claim is allowable.

**Claims 25-29** depend either directly or indirectly from claim 24 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 24, are neither disclosed nor suggested by the references of record either

1 singly or in combination with one another. In addition, as these claims are  
2 allowable, Dridi is not seen to add anything of significance to the rejection of  
3 claim 28.

4 **Claim 30** recites an Extensible Markup Language (XML) parsing system.

5 The recited system comprises:

- 6 • a collection of schema modules, each of which being configured to  
7 evaluate a different schema that is associated with an XML data  
8 stream; and
- 9 • a plurality of states associated with each schema module, individual  
10 states of a schema module defining a schema requirement relating to  
11 a particular element that is evaluated by that schema module.

12 In making out this rejection, the Office argues that Hyman discloses a  
13 collection of schema modules by virtue of the fact that Hyman discloses an XML  
14 document which has an associated schema. Applicant disagrees. Hyman does not  
15 disclose or suggest a schema module as that term is contemplated in Applicant's  
16 specification and claims. Further, the Office admits that Hyman does not disclose  
17 associating a plurality of states with each schema module. Applicant agrees. The  
18 Office then relies on Lyubashevskiy and argues that Lyubashevskiy discloses a  
19 plurality of states associated with each schema module by virtue of the fact that  
20 Lyubashevskiy discloses only that a parser is implemented as a finite state  
21 machine which preserves its states between calls.

22 Applicant submits that the Office has ascribed properties to Lyubashevskiy  
23 which it simply does not have. Based on this misinterpretation and hindsight  
24 reconstruction, the Office has rejected this claim under §103. The Office has  
25 failed to make out a prima facie case of obviousness. Accordingly, this claim is  
allowable.

1       **Claims 31-33** depend either directly or indirectly from claim 30 and are  
2 allowable as depending from an allowable base claim. These claims are also  
3 allowable for their own recited features which, in combination with those recited  
4 in claim 30, are neither disclosed nor suggested by the references of record either  
5 singly or in combination with one another. In addition, as these claims are  
6 allowable, Neumann is not seen to add anything of significance to the rejection of  
7 claims 32 and 33.

8  
9       **Conclusion**

10       All of the claims are in condition for allowance. Accordingly, Applicant  
11 requests that a Notice of Allowability be issued forthwith. If the Office's next  
12 anticipated action is to be anything other than issuance of a Notice of Allowability,  
13 Applicant requests that the undersigned be contacted for the purpose of scheduling  
14 an interview.

Version of the Specification with Markups to Show Changes

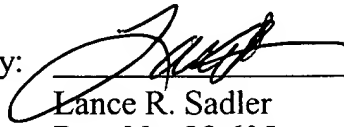
Please replace page 15, lines 7-24 with the following:

--The schema module 104 is provided as an interface with the node factory [104] 102. Schema modules can be built for each and every particular type of schema that might be expected to be received by the server. Thus, each schema module that might be built is specialized for handling one particular type of schema (or a plurality of similar schemas). The schema module knows the particular processing requirements of its associated schema, the rules that are associated with a particular schema's structure and what is necessary in order for the schema to be properly processed by the server so that an appropriate response can be returned to the client. If any of the rules for a particular schema are violated, the schema module can take appropriate action. In this example, an appropriate action might be to ignore the information that is erroneous to, and not understood by the schema module. Such action might also include generating a particular error message and returning the error message to the node factory 102. By ignoring the information that is not understood by the schema module, parsing activities are made more efficient because this information does not have to be further processed by the server in order to build and return a response to the client. The server might simply return an error message that indicates that this particular information is not understood by the server.—

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25

Respectfully submitted,

Dated: 1/24/03

By:   
Lance R. Sadler  
Reg. No. 38,605  
(509) 324-9256 ext. 226